More code, more problems

Reduce Your Risk

Plugins provide a lot of functionality but, depending on their quality, they can provide unwanted security issues as well. We show you how to protect yourself from plugins.

By Kurt Seifried

Open source software never ceases to amaze me. Of all the things I have ever encountered (with the possible exception of Legos) nothing can be so easily modified, changed, molded, extended, and otherwise improved. With many programs explicitly supporting external plugins and extensions, this process has become even easier. Some examples include Firefox, TYPO3, WordPress, and, of course, the Linux kernel.

So, why are plugins (extensions, modules, etc.) such a big deal? Plugins allow people to try out new ideas and concepts for software without having to get agreement or even cooperation from the main project (you can just run with it). Sometimes, for a variety of technical, legal, marketing, or political reasons, software projects cannot or will not provide certain features (e.g., Firefox and AdBlock Plus).

Additionally, for projects such as TYPO3 and WordPress that provide content management systems (CMS) and blog publishing capabilities (basically a specialized subset of CMS), the source code would balloon to a ridiculous size if every single capability that people wanted was included (WordPress has 11,231 plugins as of September 2010; Firefox currently appears to have more than 13,000).

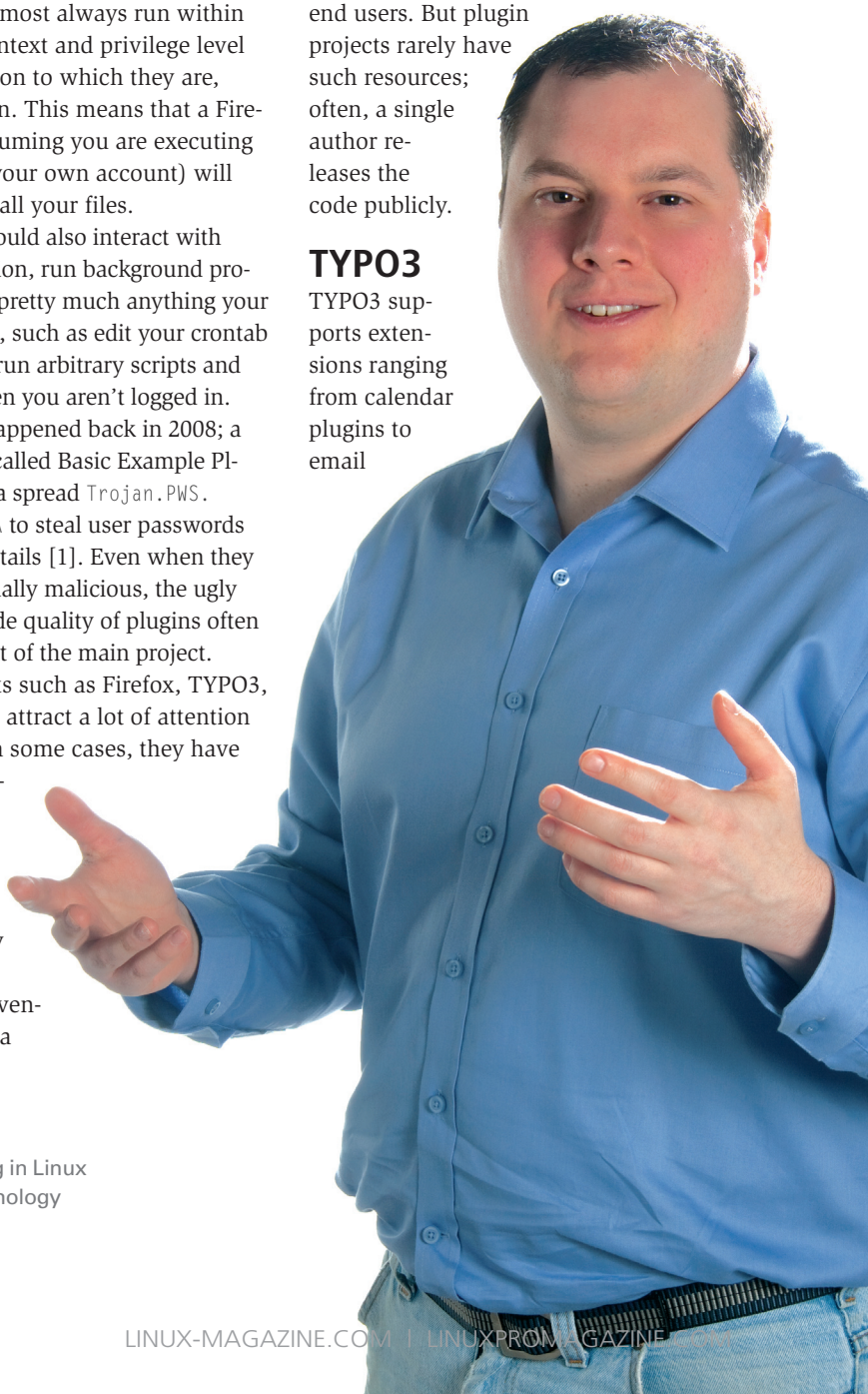
Finally, in the (relatively rare) case, an author or company feels they can only provide the software in a closed source form; in the case of the Linux kernel, for example, this allows you to have wireless drivers that otherwise would not be available.

Why Worry?

To quote Stan Lee: "With great power there must also come – great responsibility!" Plugins almost always run within the security context and privilege level of the application to which they are, well, plugged in. This means that a Firefox plugin (assuming you are executing Firefox under your own account) will have access to all your files.

The plugin could also interact with your login session, run background processes, and do pretty much anything your account can do, such as edit your crontab or `.profile` to run arbitrary scripts and code, even when you aren't logged in. This actually happened back in 2008; a Firefox plugin called Basic Example Plugin for Mozilla spread `Trojan.PWS.ChromeInject.A` to steal user passwords and account details [1]. Even when they aren't intentionally malicious, the ugly truth is that code quality of plugins often lags behind that of the main project.

Large projects such as Firefox, TYPO3, and WordPress attract a lot of attention and support; in some cases, they have dedicated security teams that continuously audit code and work to handle security issues quickly. Large security vendors also have a vested interest in finding security flaws in these programs, because their customers are likely to be using them. Fortunately, most security vendors work to ensure that security issues are handled in a manner that causes minimal pain to end users. But plugin projects rarely have such resources; often, a single author releases the code publicly.



TYPO3

TYPO3 supports extensions ranging from calendar plugins to email

KURT SEIFRIED

Kurt Seifried is an Information Security Consultant specializing in Linux and networks since 1996. He often wonders how it is that technology works on a large scale but often fails on a small scale.

workflow, a rich text editor, and various forum software packages, to name a few. The core of TYPO3 consists of more than 6,000 files of which 1,200 or so are PHP files. In 2009 (because 2010 isn't over yet, I'll use last year), there were 21 security advisories with 113 issues. Of these issues, only 16 were in the actual TYPO3 core code [2]; the other 97 were in TYPO3 extensions.

This number isn't as bad as it sounds. According to the TYPO3 Extension Repository page, the 10th most popular plugin has been downloaded 317 times for the current version, which means the majority of TYPO3 plugins have been downloaded fewer than 300 times (of course, a plugin may be downloaded and installed at more than one site, but I suspect these numbers are representative). In fact, the TYPO3 security team has labeled a large number of advisories with the text "This Collective Security Bulletin (CSB) is a listing of vulnerable extensions with neither significant download numbers, nor other special importance amongst the TYPO3 Community" [3].

According to Common Vulnerabilities and Exposures (CVE) numbers for the past few years, the TYPO3 core had 20 vulnerabilities and various extensions had 235 – a ratio roughly equal to the detailed numbers from 2009. The same story applies for Firefox, WordPress, and virtually every other software that supports plugins or extensions.

The Linux Kernel

Another great example of the use of plugins (or, in this case, modules) is the Linux kernel. On most systems running a stock vendor kernel, you will have more than 1,000 modules (e.g., CentOS 5.4 has 1,251). The total size of these modules is 96MB, meaning your kernel would go from 2MB to 98MB (i.e., your `/boot` partition would need to be a gigabyte or two). To protect themselves from binary-only modules, Linux kernel developers have implemented a system that marks the kernel as "tainted" if a module is loaded that does not have an open source license. Because the Linux kernel uses modules for most non-core functions, it is relatively quick and easy to upgrade the system without rebooting.

Short-Term Solutions

Several short-term and long-term solutions are available for managing vulnerabilities. The most obvious short-term solution is to reduce the attack surface. Firefox is a great example of this, because several applications quietly install plugins (Java, Skype, etc.) that you might not be aware of. Well-behaved extensions like NoScript or AdBlock Plus (Figure 1) will give you an "Uninstall" option; less well-behaved extensions like the Java Console (Figure 2) won't have an Uninstall option but typically have a Disable option.

Extensions that have neither a Disable nor Uninstall option should generally be avoided. But at least in Linux, if you have an unruly extension, you can manually remove it by simply going into your home directory, entering the `~/.mozilla/firefox/` directory, entering your personal default directory (it will have a random name that helps prevent attackers from dropping files in there), and deleting the directories for the extensions you want to get rid of [4]. The only problem will be figuring out which directory is which extension; they use a GUID (really long, hopefully unique string) as the directory name, such as

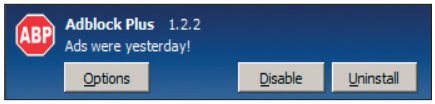


Figure 1: A well-behaved plugin (AdBlock Plus).

```
ec8030f7-c20a-464f-9b0e-13a3a9e97384
```

so you need to use the search engine of your choice to find out which directories are which plugins. The same issues generally apply to web applications. Most plugins and extensions will usually behave, but, if not, you might have to go in manually and start deleting files.

Because you can't always remove plugins that could present a threat, you can do some other things to make sure you are getting the best possible plugins. The first thing to consider is freshness: When was the last release and how often are the releases? Stale plugins are usually more dangerous; security fixes take more time, if they are ever updated. Also, check the project's website. Whether a plugin or extension has a decent website with contact information can indicate whether you want to use it (hint: no contact information is a bad sign).

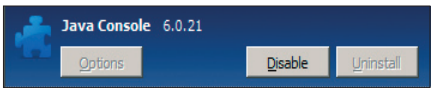


Figure 2: A less well behaved plugin (Java Console).

Long-Term Solutions

One of the best long-term solutions (potentially) is a WordPress-related project. WordPress announced that they would be separating plugins into two camps: the "core" plugins and the rest. Core plugins would be the popular plugins that everyone uses and, thus, provide the highest exposure to attackers. These core plugins will (one hopes) be audited and integrated with WordPress, resulting in higher code quality and easier installation and management. Unfortunately, not much seems to have happened with this project. (I suspect this is a classic case of "good enough" stifling an effort to improve things.)

Conclusion

Plugins and extensions are here to stay. Unfortunately, the code quality of many ranges from so-so to downright terrible. My advice is not to use any plugins that don't receive regular updates or don't behave nicely. On the other hand, certain plugins like NoScript, FlashBlock, and AdBlock Plus can significantly improve the security of your system (considering the number of Adobe Flash zero-day attacks in 2010) [5]. ■■■

INFO

[1] Malicious Firefox plugin: *http://blog.mozilla.com/security/2008/12/08/malicious-firefox-plugin/*

[2] TYPO3 security: *http://typo3.org/teams/security/security-bulletins/*

[3] TYPO3 security bulletin: *http://typo3.org/teams/security/security-bulletins/typo3-sa-2010-018/*

[4] Uninstalling add-ons: *http://kb.mozillazine.org/Uninstalling_extensions*

[5] Zero-day vulnerability: *http://en.wikipedia.org/wiki/Zero-day_attack*